Rapid Distribution of Tasks on a Commodity Grid

Ladislau Bölöni[1], Damla Turgut[1], Taskin Kocak[1], Yongchang Ji[2], and Dan C. Marinescu[2]

[1] Department of Electrical and Computer Engineering
[2] School of Computer Science,
University of Central Florida,
Orlando, FL 32816
{lboloni, turgut, tkocak}@cpe.ucf.edu,{yji,dcm}@cs.ucf.edu

Abstract. The global internet is rich in commodity resources but scarce in specialized resources. We argue that a grid framework can achieve better performance if it separates the management of commodity tasks from the tasks requiring specialized resources. We show that the performance of task execution on a commodity grid is the delay of entering into execution. This effectively transforms the resource allocation problem into a routing problem.

We present an approach in which commodity tasks are distributed to the computation service providers by the use of a forwarding mesh based on randomized Hamilton cycles. We provide stochastically weighted algorithms for forwarding. Mathematical analysis and extensive simulations demonstrate that the approach is scalable and provides efficient task allocation on networks loaded up to 95% of their capacity.

1 Introduction

The computational grid (and the internet at large) is rich in commodity resources but scarce in specialized resources. There is a large number of PC class hardware (Windows and Apple desktops, Unix and Linux workstations) with typically very low resource utilization. On the other hand, there is a scarcity of specialized resources, such as supercomputers, vector processors, specialized input and output devices and so on. Typically, the need for specialized resources is dictated by the nature of the application and, less often, by the chosen implementation.

If we look at the state of the art for distributed high performance computing, we see two different approaches:

– The computational grid community develops software which manages scarce specialized resources. Although the vision of grid computing was refined several times ([4] $\rightarrow$ [6] $\rightarrow$ [5] $\rightarrow$ [2]) the main deployment of grid applications are for projects with expensive specialized hardware. Examples of testbeds are the grid projects of the National Partnership for Advanced Computational Infrastructure (NPACI) and National Computational Science Alliance

P.M.A. Sloot et al. (Eds.): EGC 2005, LNCS 3470, pp. 721–730, 2005.

(NCSA) in the US or the European DataGrid project. The grid computing projects developed at IBM, Sun and Hewlett Packard are also largely fall in this category.
- A number of distributed computing initiatives are exploiting the abundance of commodity resources for solving highly parallelizable applications. Examples are SETI@Home [16], Folding@Home [13], the cryptographic challenges sponsored by RSA laboratories [15] or the Mersenne prime search [14]. The Berkeley Open Infrastructure for Network Computing (BOINC, [1, 12]) proposes to provide a framework more general than the SETI@Home project, which can be shared by a number of projects following this pattern of interaction. These projects, which rely on donated processor time are sometimes referred as "public computing".

Both approaches target grand challenge applications. The applications targeted by the grid computing community however, are more general than the typical public computing approaches. On the other hand, SETI@Home and the related applications are highly successful in harnessing large amount of cheap computing resources.

We note that many high performance computing workflows contain both specialized and commodity tasks. For the specialized tasks, the best thing the workflow engine can do is to queue them at the appropriate specialized providers, for instance through a system such as Condor [11]. For commodity subtasks however, this approach is not appropriate. There are a very large number of community service providers (on the order of millions), which makes it difficult to deploy any kind of centralized distribution system.

We note that if a task is executed on a commodity hardware, the main determining factor of the termination time is the time at which the task is taken into execution. Furthermore, given the abundance of the commodity resources, it is likely that if a task needs to be queued at a certain host, it is almost sure that somewhere on the internet there is a task which can take it into execution immediately. Under this assumption, the task allocation problem is reduced to a specialized routing problem. A similar idea is proposed in [7, 3]. The Wire Speed Grid Project at the University of Chalmers [17], proposes an architecture in which the task allocation is performed in a hardware accelerated manner on the network routers. As our tasks have a relatively long execution time, an application layer implementation would provide the same benefits.

2 Commodity Components in Grand Challenge Applications

Grand challenge applications range from the application of relatively simple algorithms on massive amounts of data (such as the SETI@Home project), to exhaustive search of a complex combinatorial problems with small amounts of input and output data (e.g. cryptographic analysis). Many of the high performance applications however, are what we call *grid workflows*. Problems with